

TECH+ スペシャルセミナー

先進企業から学ぶ最新クラウド活用事例

～DXを加速！今始める攻めと守りの自動化2.0～



DXの必需品 「攻めと守りの自動化」とは

NEC サービス&プラットフォームSI事業部

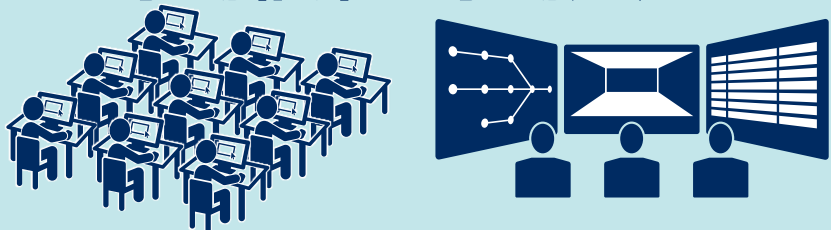
吉田 功一

新技術を取り込みにくい

密結合で保守しづらいアプリ

従来のスタティックな環境

手動構築・手動運用



モノリシック
(物理機器)

仮想化技術がまだ普及していなかった頃…

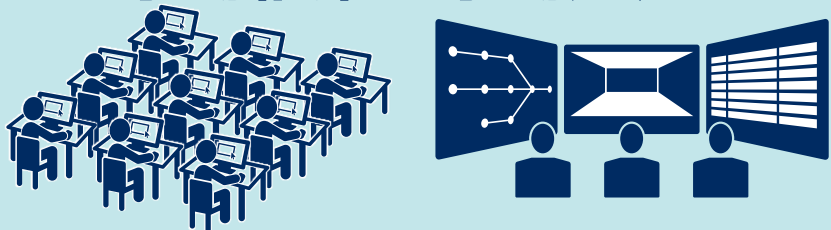
- ✓ サービス停止させないために
「**最繁トラヒック×バッファ×冗長2倍**」
のリソースを物理的に並べていた
- ✓ また「CPU、Mem、I/O」の選択肢が少
なく、**リソースを使いきれなかった**
- ✓ それでもなるべく少ない機器点数で多くの
サービスを捌けるように、**複数機能をうまく
混載(コンソリ)することが“正しい”**とされ
ていた
- ✓ **アプリケーションもこれらを前提**に作られ
ていた(それが正しかった)

新技术を取り込みにくい

密結合で保守しづらいアプリ

従来のスタティックな環境

手動構築・手動運用



モノリシック
(クラウドリフト)

仮想化技術という武器を手に入れた結果

- ✓ 「CPU、Mem、I/O」が**選択可能**に
- ✓ 「オートスケールWeb」くらいの**簡単な動的システム**は組めるようになった
- ✓ 結果、システムサイズはコンパクトになり**ハードウェア寿命から脱却**できた
- ✓ **複数機能を混載(コンソリ)**させないことが**“正しい”**と変わり始めた

しかし、以下の課題は解決されず...

- ✓ アプリケーションを作り直さない限り**多くの機能で冗長2倍は継続**
→ クラウドシフトへの期待 ... ①
- ✓ 根本的にはシステム形状は変わらないので**運用の手間は大きくは変わらない**
→ 運用自動化への期待 ... ②

新技术を取り込みにくい

密結合で保守しづらいアプリ

従来のスタティックな環境

手動構築・手動運用



モノリシック
(クラウドリフト)

クラウドシフトへの期待(①)

OPEXの効率化

アプリ設計の段階から自動化を組み込むことで、必要な時に必要なリソースを使う「**本格的な動的システム**」
(リソースの冗長確保からの脱却)

ROI(投資利益率)の引き上げ

アプリ設計の段階から自動化を組み込むことで、運用しながら機能追加できる「**廃棄容易なシステム**」(DevOpsの実現)

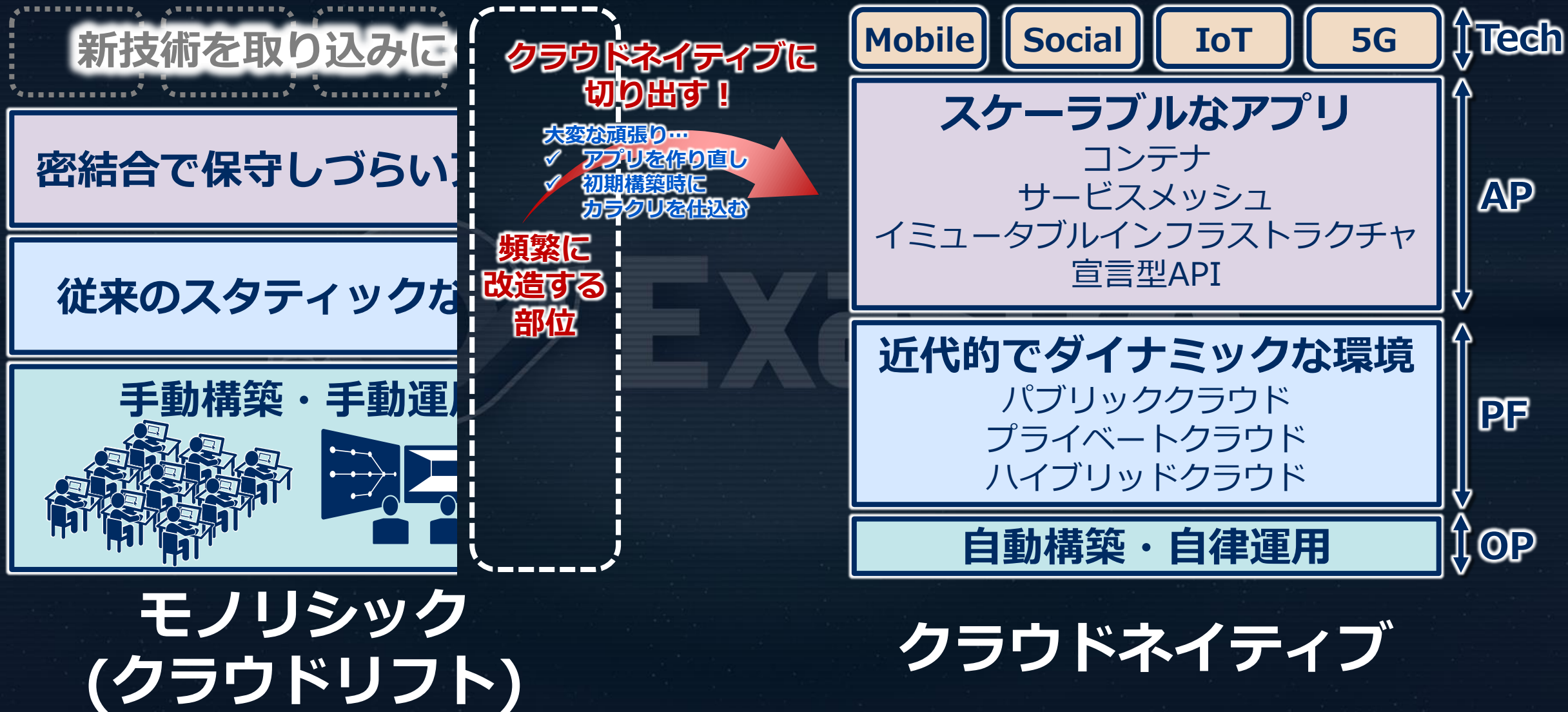
クラウドシフトの**問題点**

- ✓ アプリの作り直しを伴う
- ✓ 初期構築時にカラクリを仕込むのが大変

⇒とにかく敷居が高い！

頻繁に改造したい部位を選択しなければ…

Introduction 「攻めの自動化」 「守りの自動化」 とは？



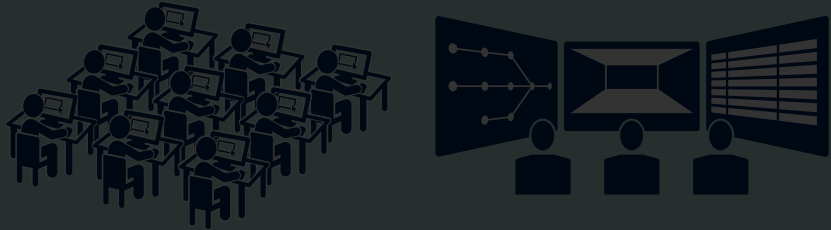
Introduction 「攻めの自動化」 「守りの自動化」 とは？

新技术を取り込みにくい

密結合で保守しづらいアプリ

従来のスタティックな環境

手動構築・手動運用



モノリシック

(物理機器クラウドリフト)

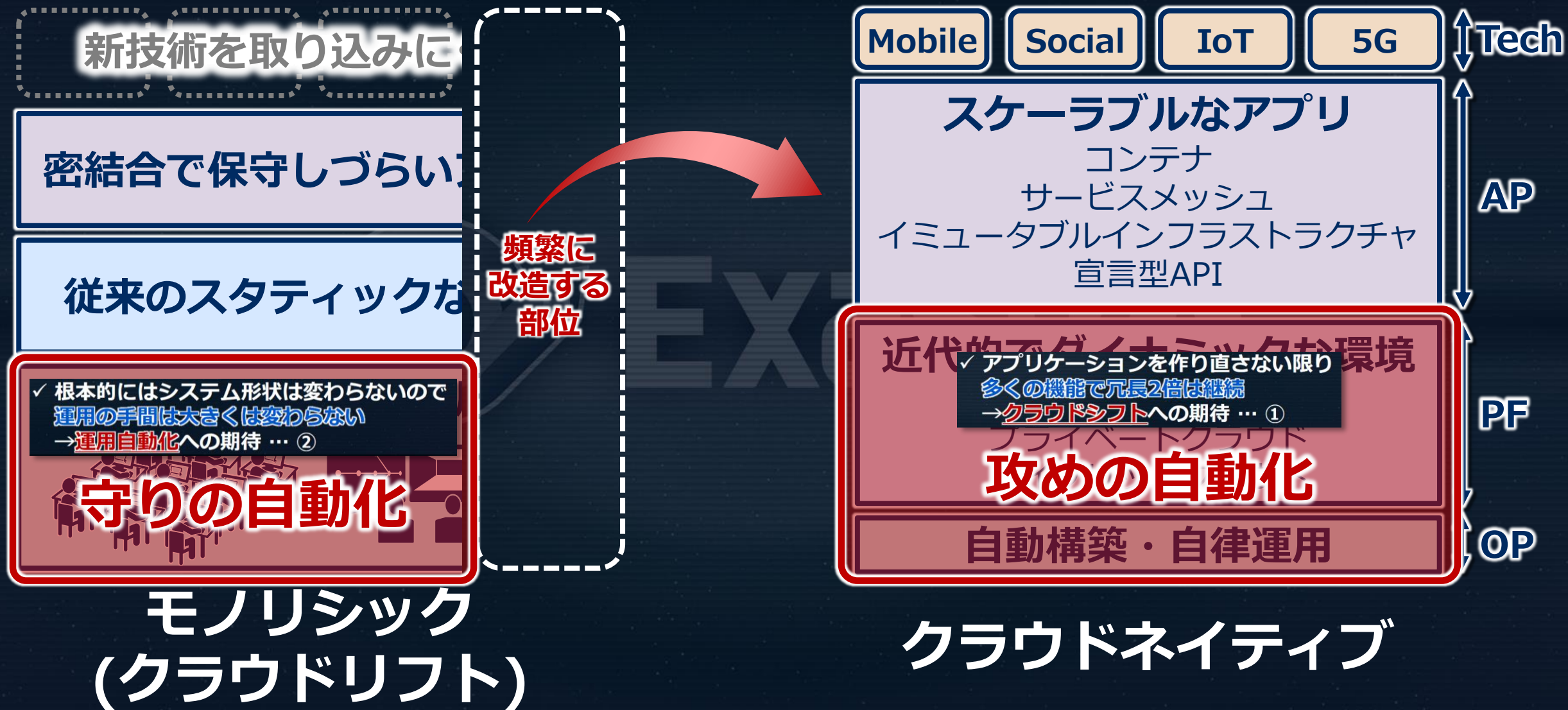
仮想化技術という武器を手に入れた結果

- ✓ 「CPU、Mem、I/O」が**選択可能**に
- ✓ 「オートスケールWeb」くらいの**簡単な動的システム**は組めるようになった
- ✓ 結果、システムサイズはコンパクトになり**ハードウェア寿命から脱却**できた
- ✓ **複数機能を混載(コンソリ)**させないことが**“正しい”**と変わり始めた

しかし、以下の課題は解決されず...

- ✓ アプリケーションを作り直さない限り**多くの機能で冗長2倍は継続**
→クラウドシフトへの期待 ... ①
- ✓ 根本的にはシステム形状は変わらないので**運用の手間は大きくは変わらない**
→運用自動化への期待 ... ②

Introduction 「攻めの自動化」 「守りの自動化」とは？



「攻めの自動化」の進め方

ところで...

クラウドネイティブの**カラクリ**とは？
初期構築時にどんな**カラクリ**を仕込んでおけば、
気持ちよくDevOpsをまわせる？

クラウドシフトで期待できること

11の**カラクリ**
(**10の技** + 仮想化)

クラウドネイティブの実現手段

最終的な目的
ITリソースを活用し
利益を向上する

ROI(投資利益率)の引き上げ

投資の質を上げて
売上につなげる

単位時間あたりの
提供機能数 = 価値の向上

高速・スケーラブルな
機能提供 (リリース)

OPEXの効率化

OPEX(運用費)を
下げて投資にまわす

運用に必要な
人件費の削減

運用に必要な
保守費の削減

CI(継続的インテグレーション)

CD(継続的デリバリー)

マイクロサービスアーキテクチャ

サービスメッシュ

ブルーグリーンデプロイメント
(イミュータブルインフラストラクチャ)

宣言的API

オートスケーリング

分散トレーシング

メトリクス監視

メンテナンスタスクの一元管理・自動化

仮想化(ハードウェア寿命からの解放)

「攻めの自動化」の進め方

クラウドシフトで期待できること

11のクラウド (10の技+仮想化)

クラウドネイティブの実現手段

ROI(投資利益率)の引き上げ

投資の質を上げて
売上につなげる

単位時間あたりの
提供機能数 = 価値の向上

高速・スケーラブルな
機能提供 (リリース)

CI(継続的インテグレーション)

CD(継続的デリバリー)

マイクロサービスアーキテクチャ

サービスメッシュ

ブルーグリーンデプロイメント
(イミュータブルインフラストラクチャ)

宣言的API

OPEXの効率化

OPEX(運用費)を
下げて投資にまわす

**特にこの6つの手段が
「ROI(投資利益率)の引き上げ」という
クラウドネイティブ最大のテーマに直結する**

オートスケーリング

分散トレーニング

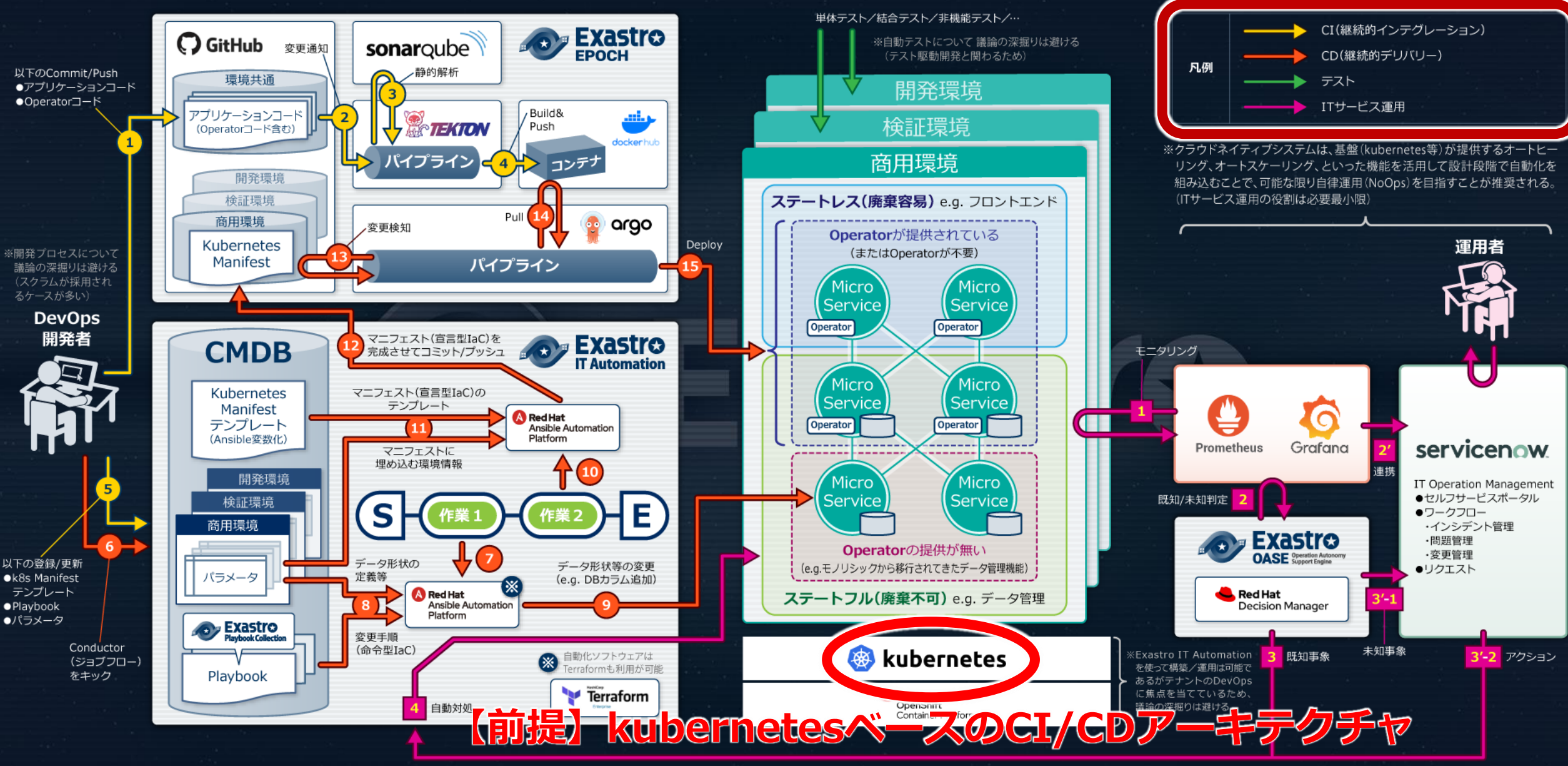
保守費の削減

仮想化(ハードウェア寿命からの解放)

最終的な目的

ITリソースを活用し
利益を向上する

「攻めの自動化」の進め方



「攻めの自動化」の進め方



「攻めの自動化」の進め方

CD(継続的デリバリー)の前半

POINT

ステートフル&Operatorの提供もないような、いわゆる「なんちゃってコンテナ」がある場合、K8sマニフェストに従ったデプロイメントよりも前に、レガシーな手段でリリース作業せざるを得ない。そういったコンテナをなるべく排除するのと同時に、手動作業で更改することは避けたほうが良い。(自動化のジョブフローに組み込むことがお薦め)

以下のCommit/Push
●アプリケーションコード
●Operatorコード

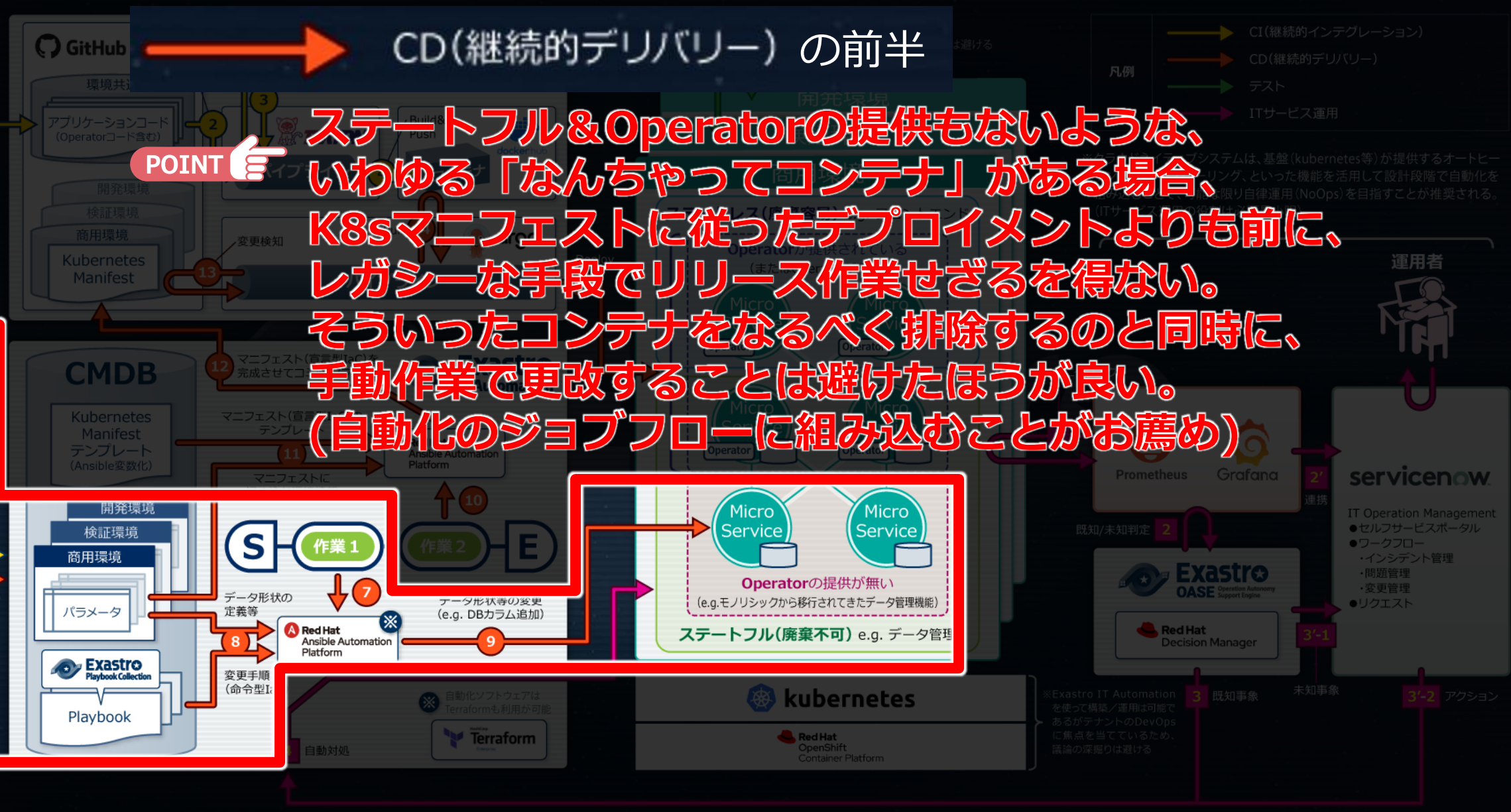
凡例
→ CI(継続的インテグレーション)
→ CD(継続的デリバリー)
→ テスト
→ ITサービス運用

※開発プロセスについて議論の深掘りは避ける(スクラムが採用されることが多い)

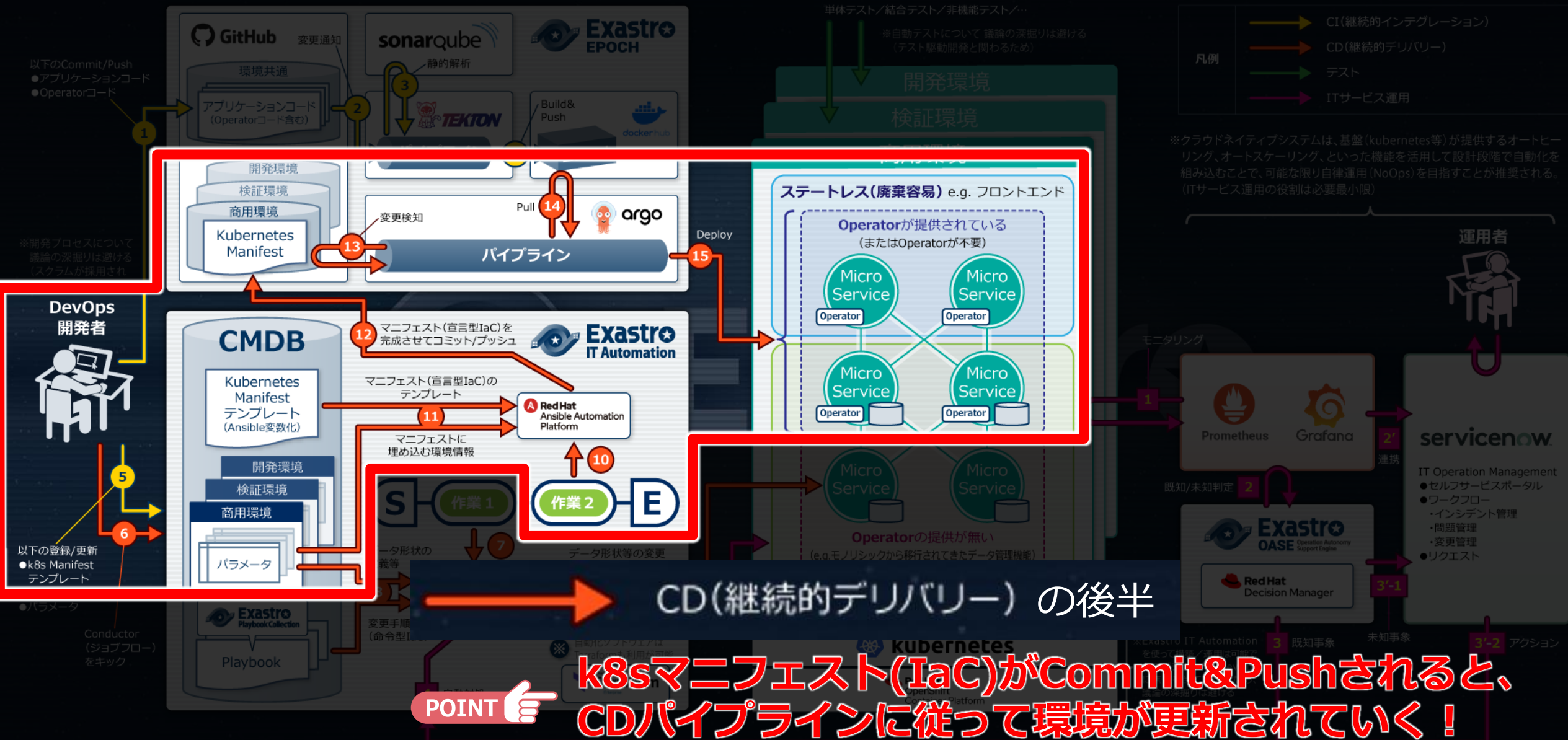


以下の登録/更新
●k8s Manifest テンプレート
●Playbook
●パラメータ

Conductor (ジョブフロー) をキック



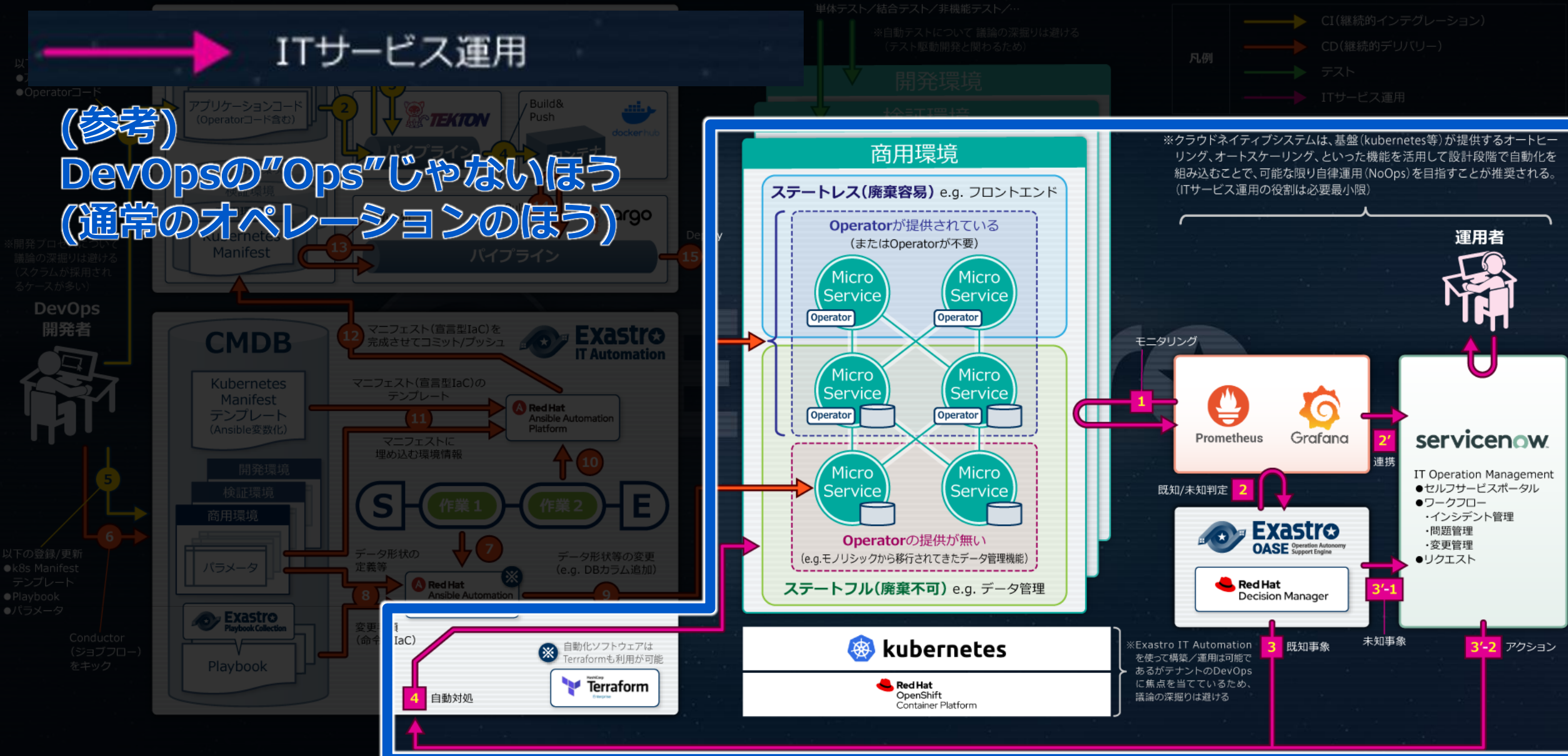
「攻めの自動化」の進め方



「攻めの自動化」の進め方

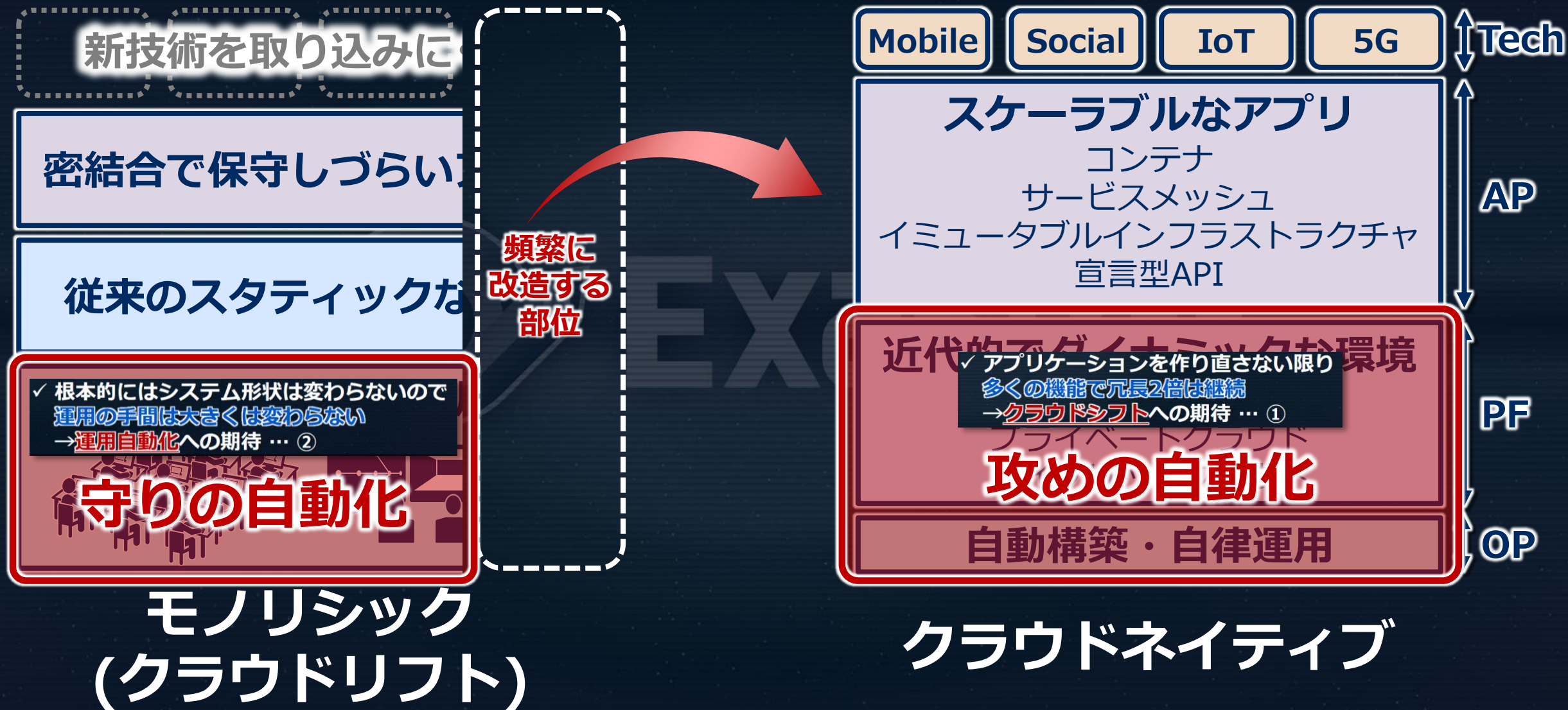
ITサービス運用

(参考) DevOpsの"Ops"じゃないほう
(通常のオペレーションのほう)

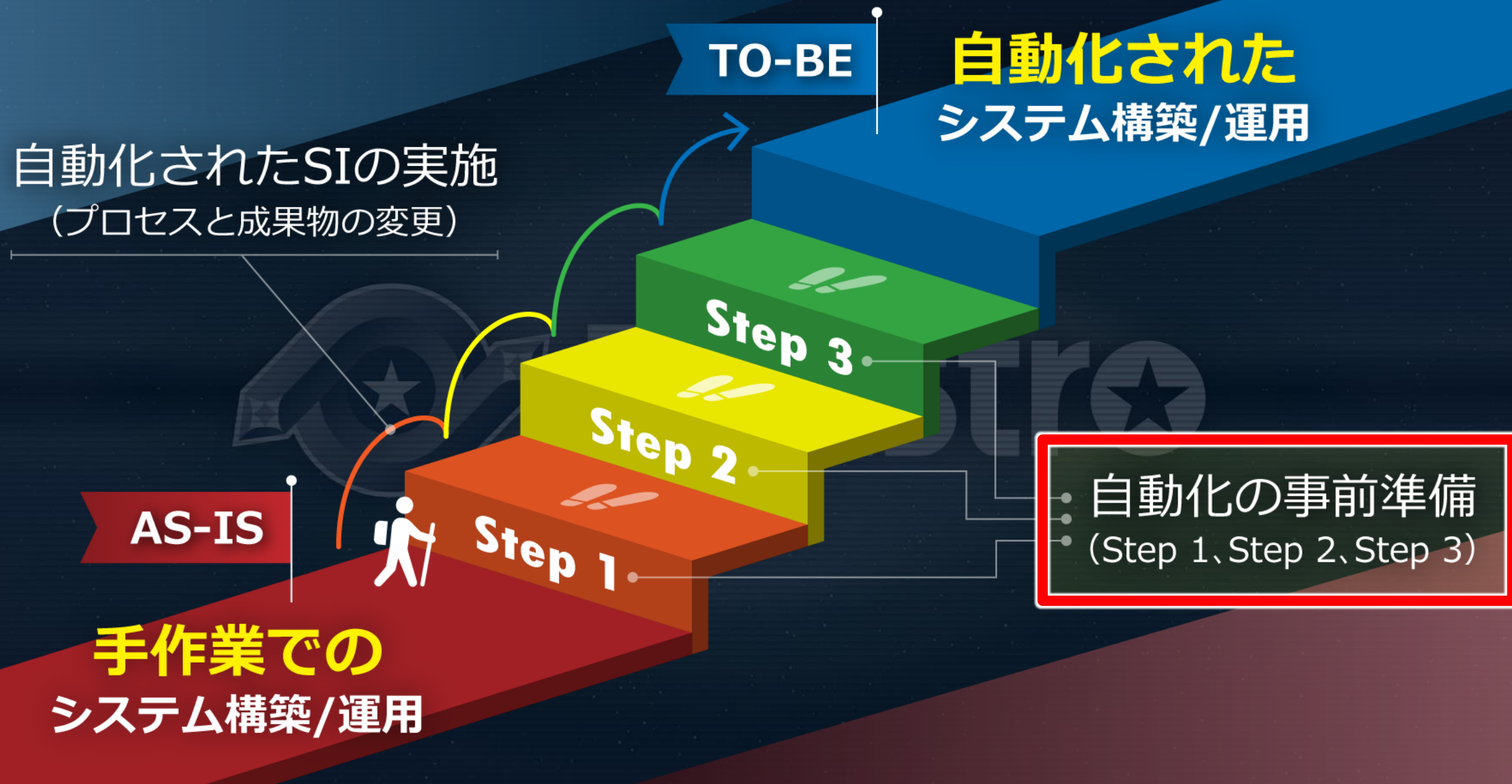


「守りの自動化」の進め方

Introduction 「攻めの自動化」 「守りの自動化」とは？



「守りの自動化」の進め方



モノリシックシステムの

構築・運用に携わるITエンジニアの現場の声

設計

- ✓ チーム間の情報伝達に遅延やミスが発生する
- ✓ データの二重管理や独自文言が設計ミスにつながる
- ✓ 多重開発により設計書(帳票)の管理が煩雑化する
- ✓ 結果として設定の前後性を確認できない

作業準備

- ✓ チーム間の作業順序が複雑で毎回タイムチャートを作成しては使い捨てる
- ✓ 作業ごとに手順書を作成/レビューしては使い捨てる
- ✓ 手順ごとにコンフィグを埋め込んでいて、新機種/新OSを追加するごとに手順書のパターンが増える(マルチベンダー対応の障壁)

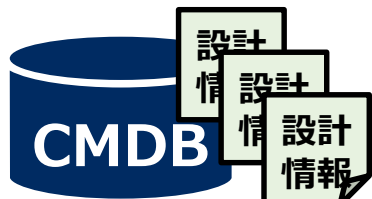
作業実施

- ✓ 人手作業なので作業時間が一定でない
⇒チーム間で作業待ちが発生
- ✓ 人手作業なので人為ミスの懸念から逃れられない

これらの課題は大まかには3つのステップで解決できます



Step 1
設計情報の一元管理

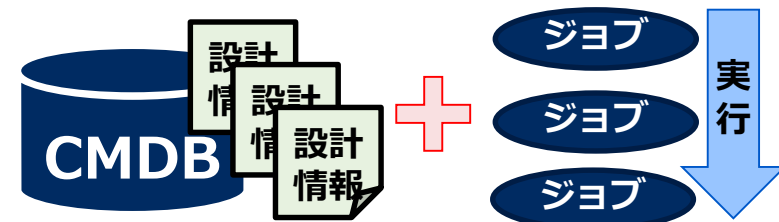


Step 2
自動実行の実現



連携

Step 3
一元管理と自動実行の連携



細かく刻めば12つのタスクで解決できます

Step 1 → Step 2 → Step 3

Step 1: 設計情報の一元管理

実施するタスク

- 設計情報の管理版の収集
- 設計情報の正規化
- Exastro IT Automation (CMDB)の構築
- CMDBへの情報投入
- CMDBの活用

タスクの説明

各チームの代表者は、自チームの設計情報を収集して、他のチームの代表者と共有する

POINT

- ① 目的を明確にして、一元管理の範囲を決める
- ② 既存の設計情報の管理方法は多様である
- ③ 事例 ~ 実際のプロジェクトで収集した設計情報

詳細 次頁

5つのタスク

Step 2: 自動実行の実現

実施するタスク

- 自動化対象となる作業の分類
- 作業の詳細化
- Ansible資料準備 (Playbook等)
- ジョブフロー構築 (Symphony)
- ジョブフロー実行 (Symphony)

タスクの説明

手作業で実施している作業を整理して、自動化する作業を選定します。整理が複数のチームをまたぐ場合は、各チームの代表者が調整します。

POINT

- ① 「丁度よい」 粒度で作業を分類する
- ② 作業の効果を見積もって、優先度をつける

詳細 次頁

5つのタスク

Step 3: 一元管理と自動実行の連携

実施するタスク

- 変数と員体値の紐づけ
- ジョブフロー実行 (Symphony)

タスクの説明

Exastro IT Automationの「代入値自動登録設定」を利用して、パラメータシートとの値とジョブ内の変数とを紐づける

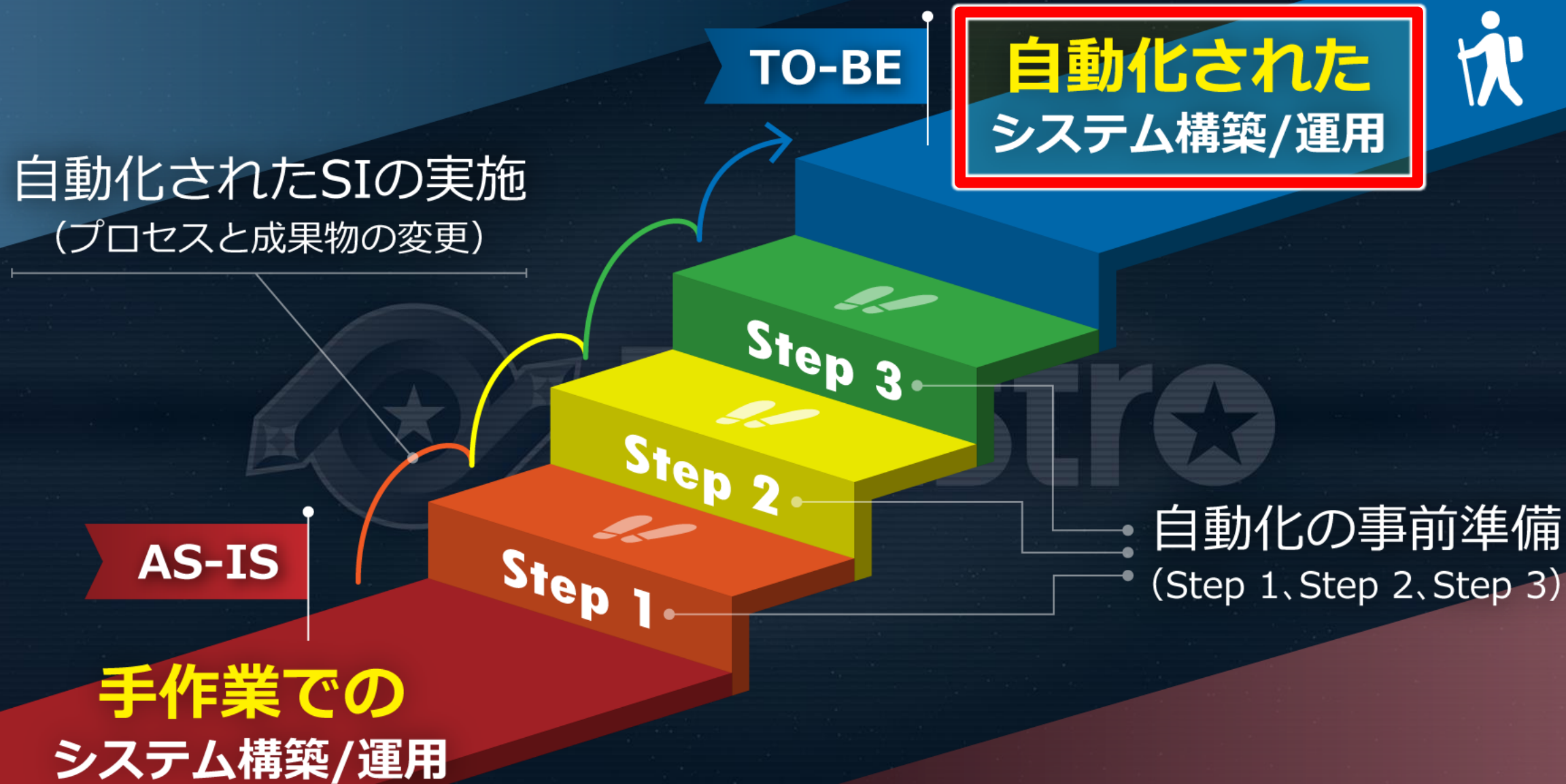
POINT

- ① Valueタイプの活用方法
- ② Keyタイプの活用方法
- ③ Key-Valueタイプの活用方法

詳細 次頁

2つのタスク

「守りの自動化」の進め方



「守りの自動化」の進め方

フェーズにおけるQCDの変化

凡例： 😊 変化なし 😄 改善 🤖 追加の検討項目あり

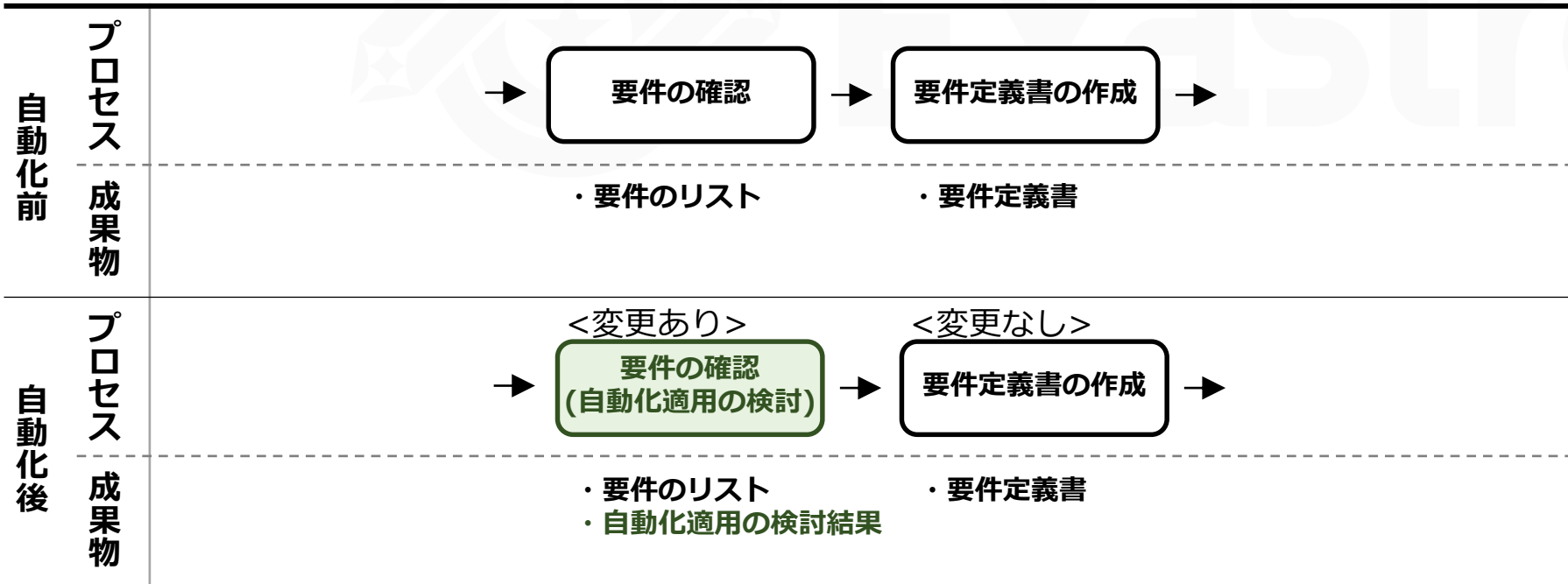
	要件定義			基本設計			詳細設計			運用設計			製造			テスト			リリース					
	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D			
自動化前	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊
自動化後	😊	🤖	🤖	😊	😊	😊	😄	😄	😄	😊	😊	😊	😄	😄	😄	😊	😊	😊	😄	😄	😄	😊	😊	😊

解説

要件の確認の段階で、自動化の適用範囲等について検討を行い、合意を取る必要がある。その検討分としてCとDが増加する。

プロセスと成果物の変化

凡例： 変更なし 作業名 変更あり 作業名 追加 作業名 消滅 作業名



「守りの自動化」の進め方

フェーズにおけるQCDの変化

凡例： 😊 変化なし 🌟 改善 🧐 追加の検討項目あり

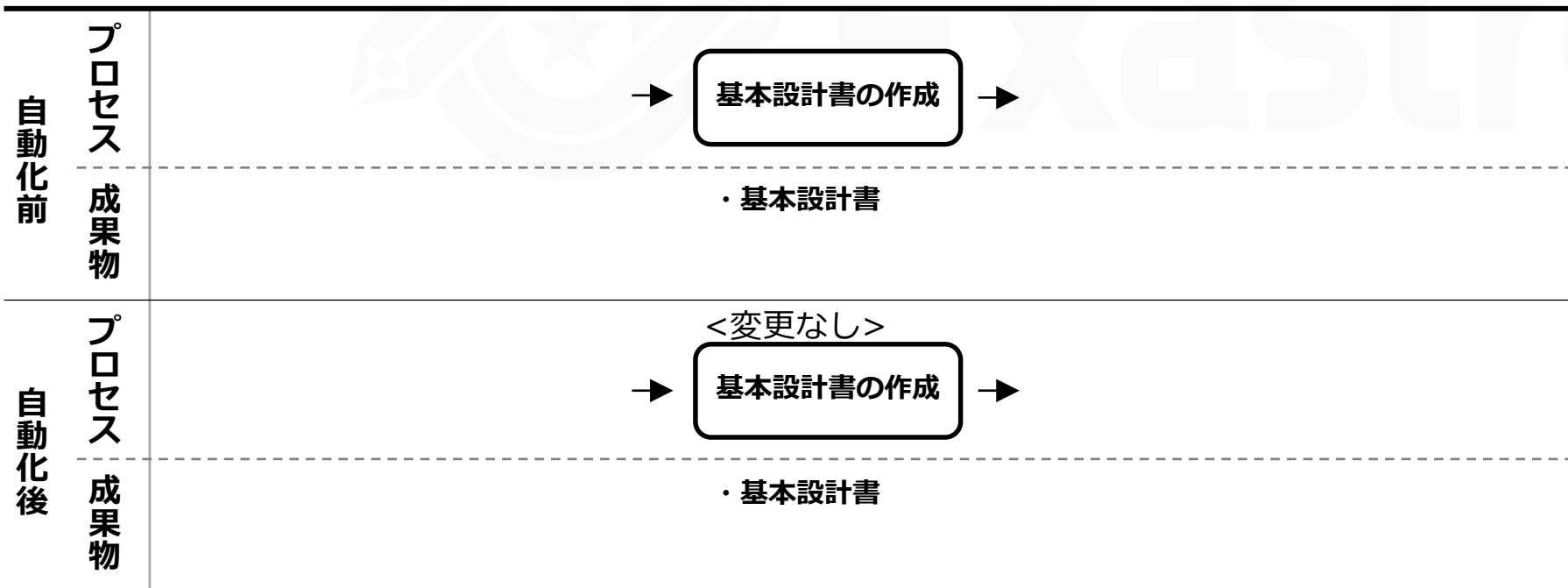
	要件定義			基本設計			詳細設計			運用設計			製造			テスト			リリース					
	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D			
自動化前	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊
自動化後	😊	🧐	🧐	😊	😊	😊	🌟	🌟	🌟	😊	😊	😊	🌟	🌟	🌟	😊	😊	😊	🌟	🌟	🌟	😊	😊	😊

解説

基本設計に取り込むべき内容は、自動化の事前準備の段階で検討済みであるため、ここでは追加の作業はない。

プロセスと成果物の変化

凡例： 変更なし 作業名 変更あり 作業名 追加 作業名 消滅 作業名



「守りの自動化」の進め方

フェーズにおけるQCDの変化

凡例： 😊 変化なし 😄 改善 🧐 追加の検討項目あり

	要件定義			基本設計			詳細設計			運用設計			製造			テスト			リリース		
	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D
自動化前	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊
自動化後	😊	🧐	🧐	😊	😊	😊	😄	😄	😄	😊	😊	😊	😄	😄	😄	😊	😊	😊	😄	😄	😄

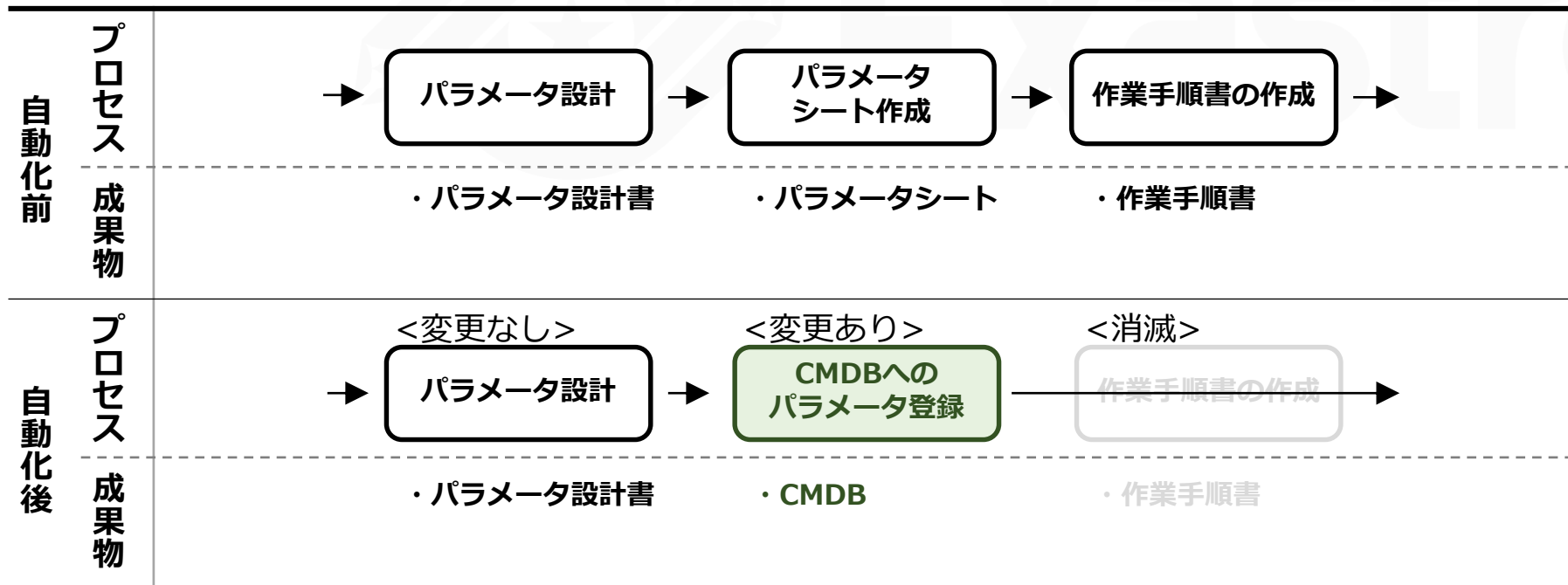
解説

パラメータ設計で作成された各パラメータは、CMDBに登録していく。これによりパラメータの形式化が進み、表記ゆれや曖昧さが提言するため、Qが改善する。

また、パラメータの適用順序などの作業手順は、自動化の事前準備の段階で作成したジョブフローに置き換わるため、作業手順書の作成のタスクが消滅する。これにより、CとDも改善する。

プロセスと成果物の変化

凡例： 変更なし [作業名] 変更あり [作業名] 追加 [作業名] 消滅 [作業名]



「守りの自動化」の進め方

フェーズにおけるQCDの変化

凡例： 😊 変化なし 🟡 改善 🧐 追加の検討項目あり

	要件定義			基本設計			詳細設計			運用設計			製造			テスト			リリース		
	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D
自動化前	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊
自動化後	😊	🧐	🧐	😊	😊	😊	🟡	🟡	🟡	😊	😊	😊	🟡	🟡	🟡	😊	😊	😊	🟡	🟡	🟡

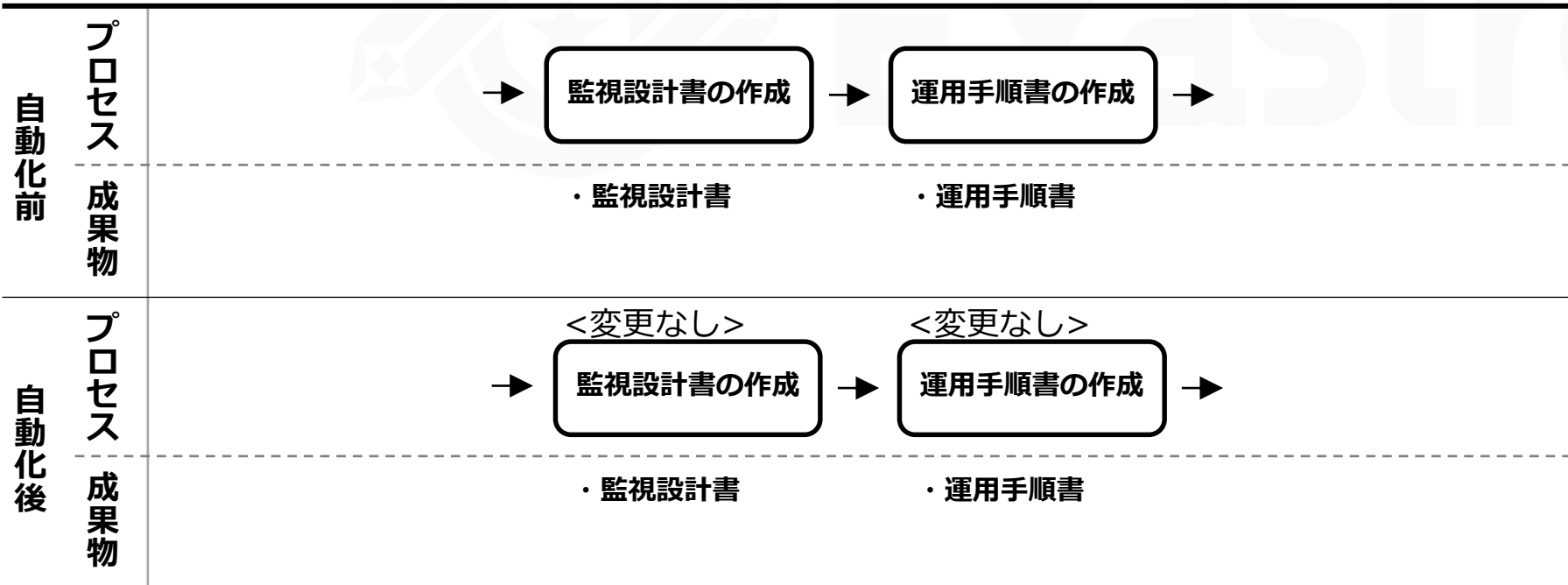
解説

本書は構築の自動化に焦点を当てているため、運用の自動化は範囲外とする。

運用の自動化を実施する場合は、QCDおよびプロセスの変化が発生する。

プロセスと成果物の変化

凡例： 変更なし 作業名 変更あり 作業名 追加 作業名 消滅 作業名



「守りの自動化」の進め方

フェーズにおけるQCDの変化

凡例： 😊 変化なし 🟡 改善 🧐 追加の検討項目あり

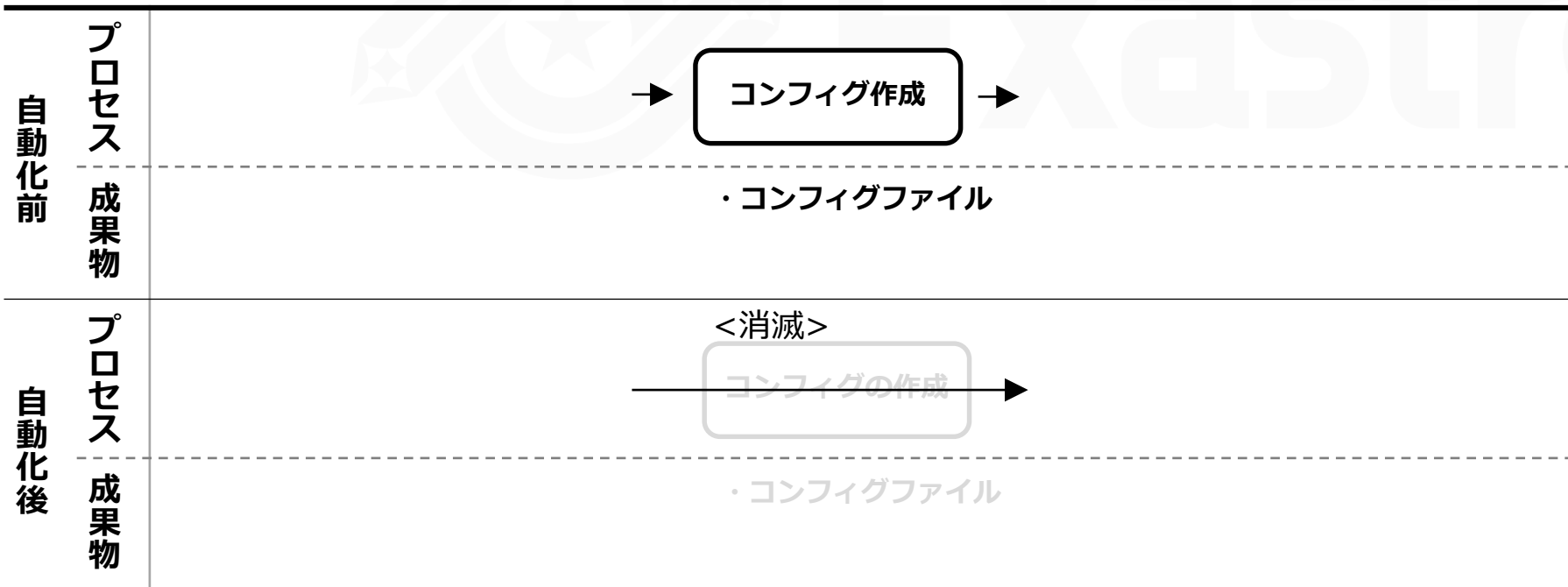
	要件定義			基本設計			詳細設計			運用設計			製造			テスト			リリース					
	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D			
自動化前	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊
自動化後	😊	🧐	🧐	😊	😊	😊	🟡	🟡	🟡	😊	😊	😊	🟡	🟡	🟡	😊	😊	😊	🟡	🟡	🟡	🟡	🟡	🟡

解説

詳細設計に基づき作成されるコンフィグファイルは、IaCとCMDBにより自動生成されるため、コンフィグ作成のタスクは消滅する。

プロセスと成果物の変化

凡例： 変更なし 作業名 変更あり 作業名 追加 作業名 消滅 作業名



「守りの自動化」の進め方

フェーズにおけるQCDの変化

凡例： 😊 変化なし 🌟 改善 🧐 追加の検討項目あり

	要件定義			基本設計			詳細設計			運用設計			製造			テスト			リリース					
	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D			
自動化前	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊
自動化後	😊	🧐	🧐	😊	😊	😊	🌟	🌟	🌟	😊	😊	😊	🌟	🌟	🌟	😊	😊	😊	😊	😊	😊	🌟	🌟	🌟

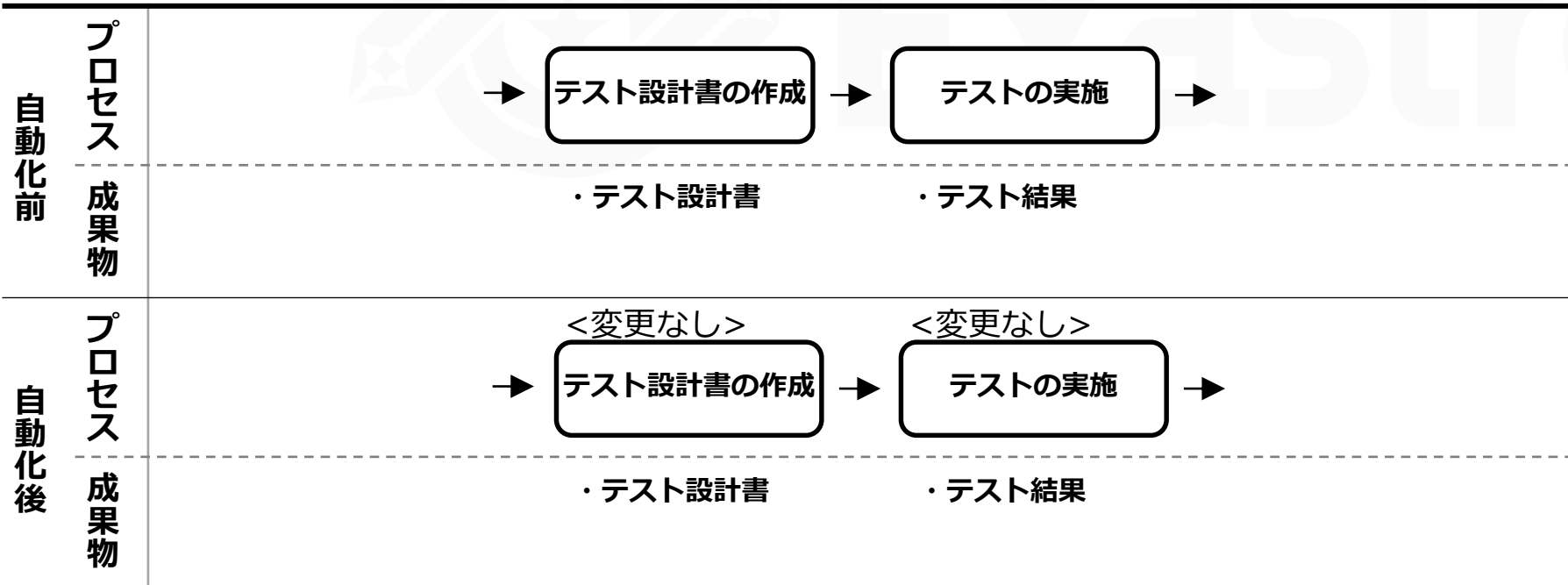
解説

本書は構築の自動化に焦点を当てているため、テスト自動化は範囲外とする。

テストの自動化を実施する場合は、QCDおよびプロセスの変化が発生する。

プロセスと成果物の変化

凡例： 変更なし [作業名] 変更あり [作業名] 追加 [作業名] 消滅 [作業名]



「守りの自動化」の進め方

フェーズにおけるQCDの変化

凡例： 😊 変化なし 😄 改善 🧐 追加の検討項目あり

	要件定義			基本設計			詳細設計			運用設計			製造			テスト			リリース					
	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D	Q	C	D			
自動化前	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊	😊
自動化後	😊	🧐	🧐	😊	😊	😊	😄	😄	😄	😊	😊	😊	😄	😄	😄	😊	😊	😊	😄	😄	😄	😄	😄	😄

解説

詳細設計で作成したジョブフローの実行がメインの作業になる。

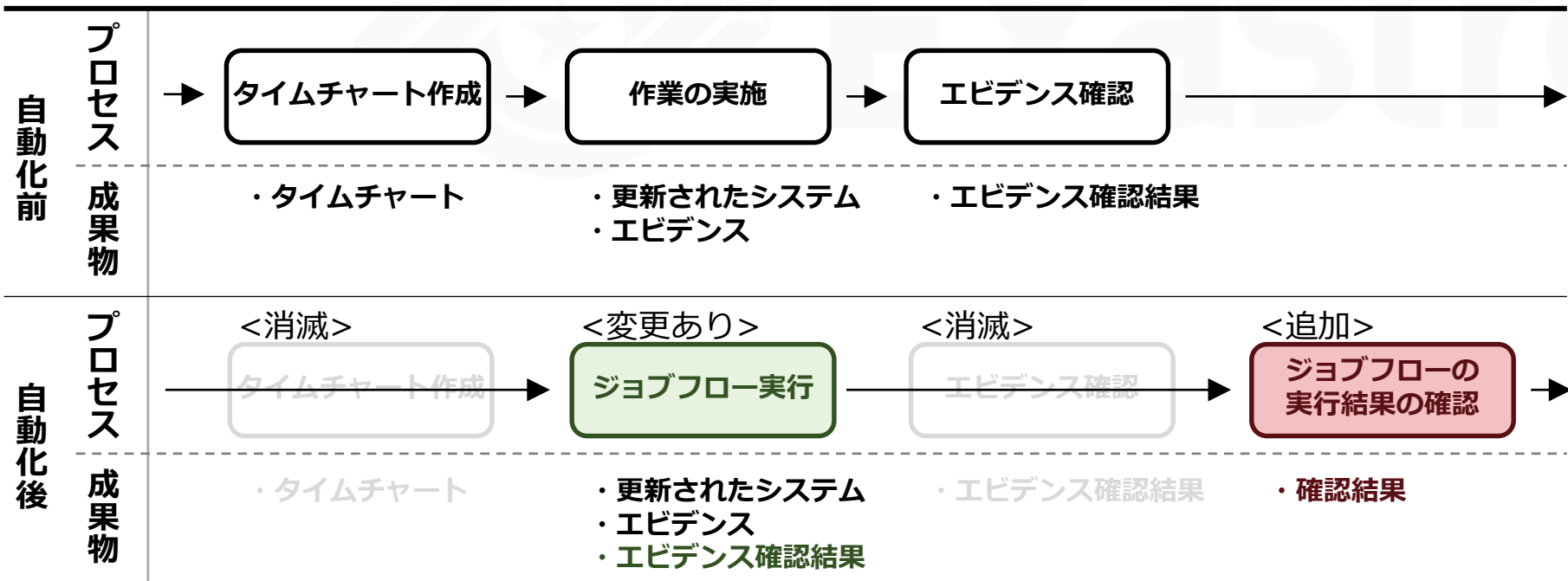
タイムチャート作成がジョブフロー実行に置き換わるため消滅。

またジョブフロー中でエビデンスの確認も実施するため、タスクとしてのエビデンス確認も消滅。

よって、ジョブフローの実行と結果の確認のみとなり、QCDともに改善する

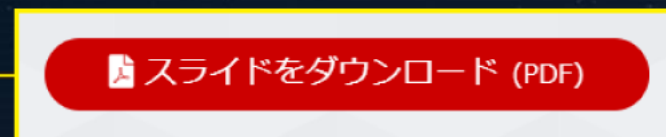
プロセスと成果物の変化

凡例： 変更なし [作業名] 変更あり [作業名] 追加 [作業名] 消滅 [作業名]



ガイドはOSSコミュニティサイトからダウンロードできます

Exastro IT Automation コミュニティサイト https://exastro-suite.github.io/it-automation-docs/index_ja.html

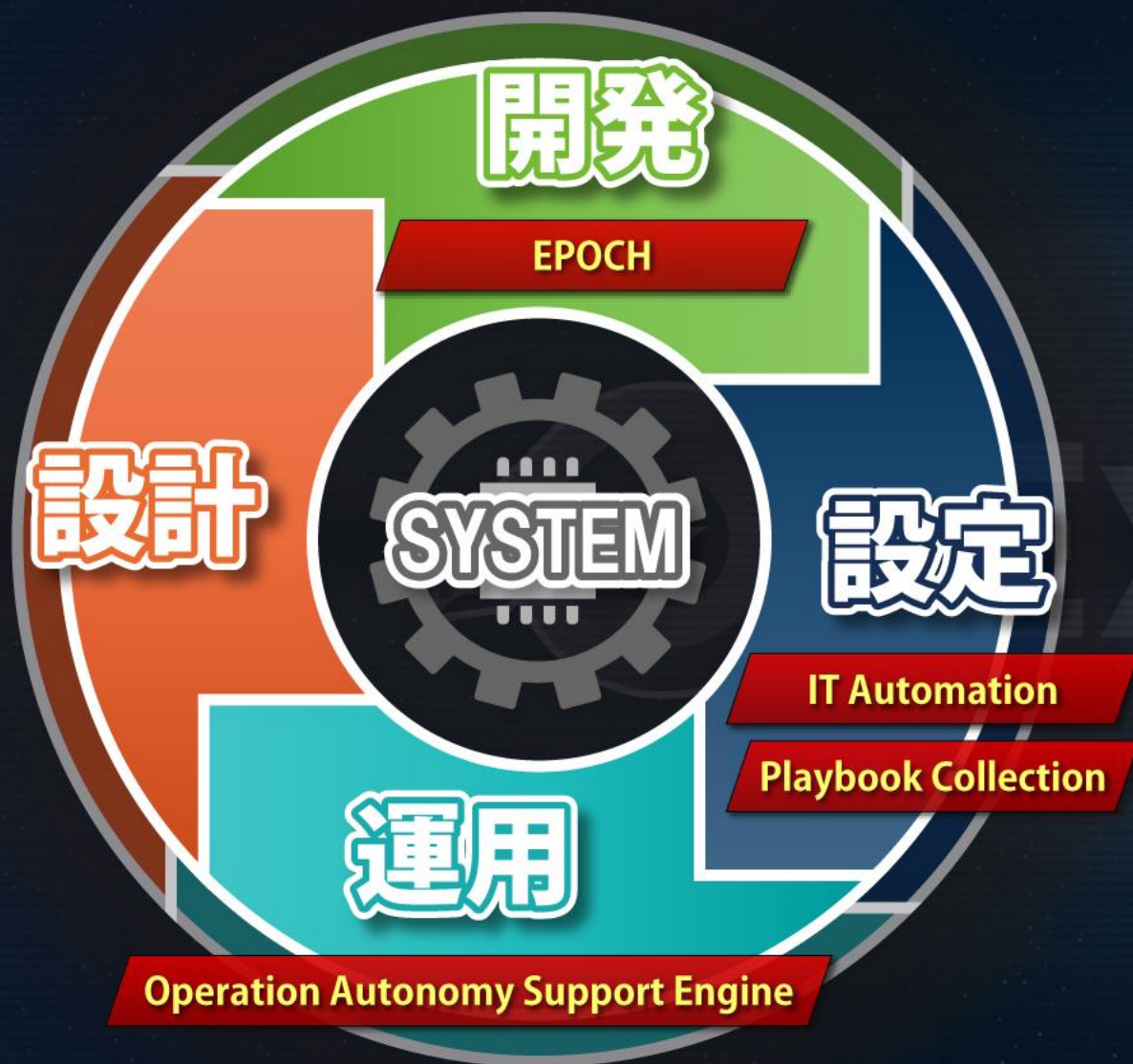


Exastro Suite

- 「攻めと守りの自動化」の実現手段 -



Exastro Suiteとは？



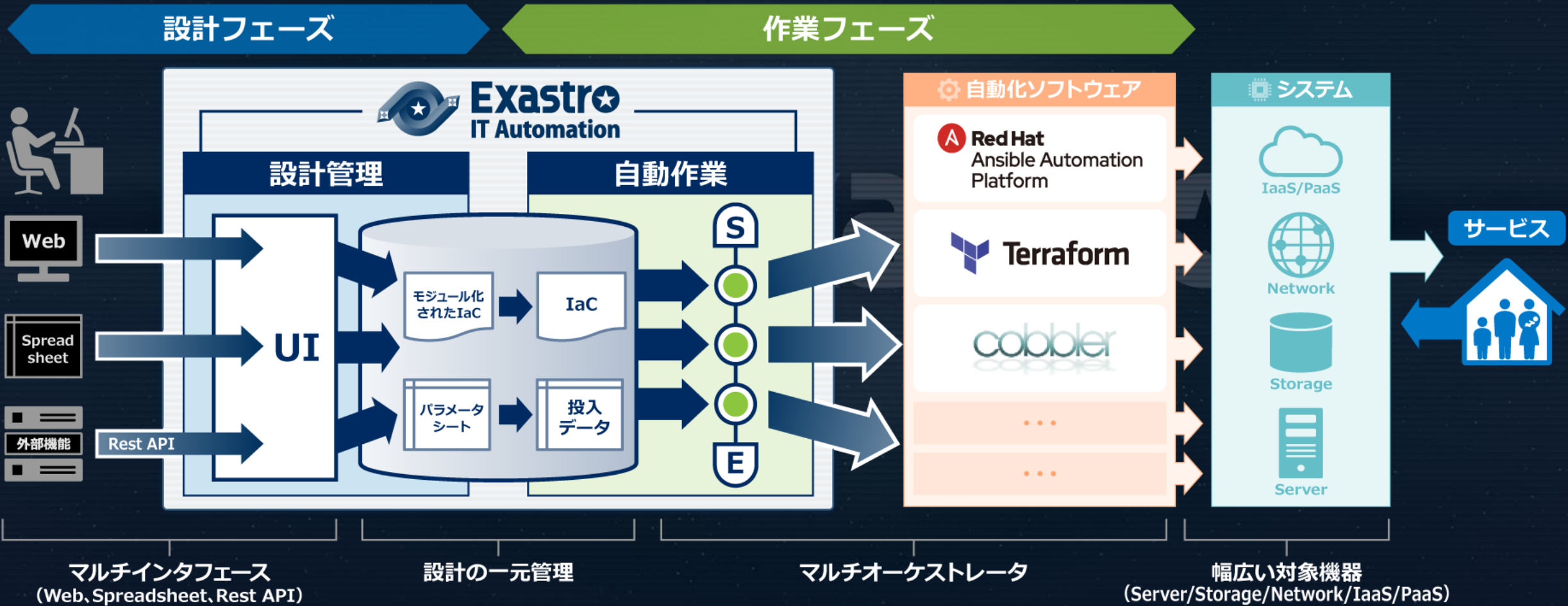
Exastro はシステムライフサイクル



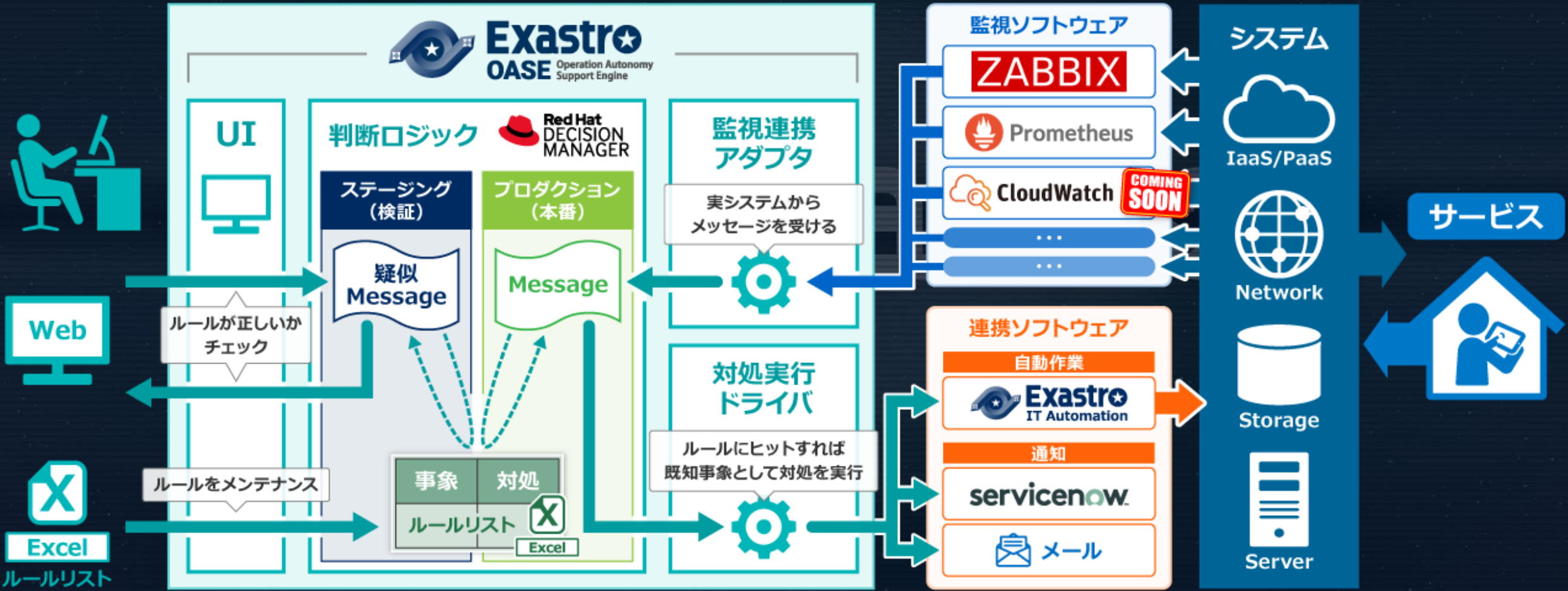
を

デジタル化・自動化・省力化することを目的とした
オープンソースのソフトウェアスイートです。

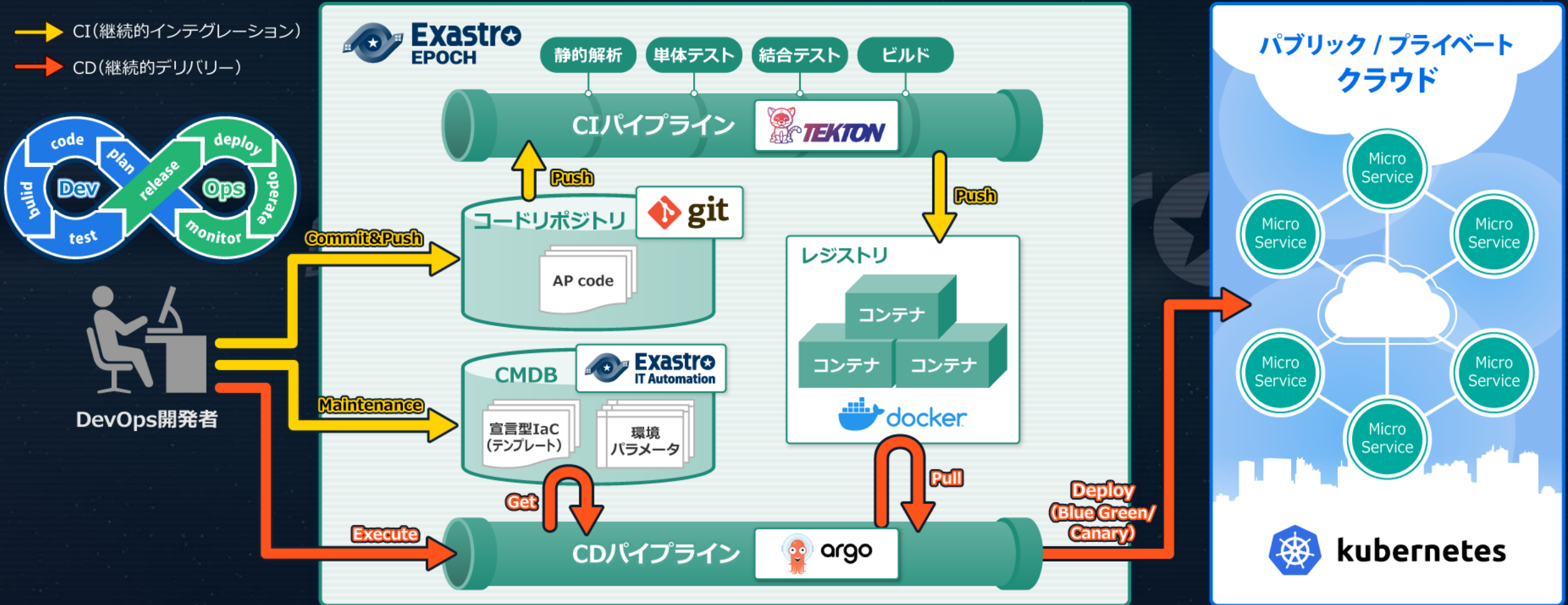
Exastro IT Automationは「システム構成(IaC+パラメータ)を管理するためのフレームワーク」です



Exastro Operation Autonomy Support Engineは「システム運用の自動化を支援するためのフレームワーク」です



Exastro EPOCHは「クラウドネイティブシステム開発を加速するためのフレームワーク」です



詳しくはコミュニティサイトへ！



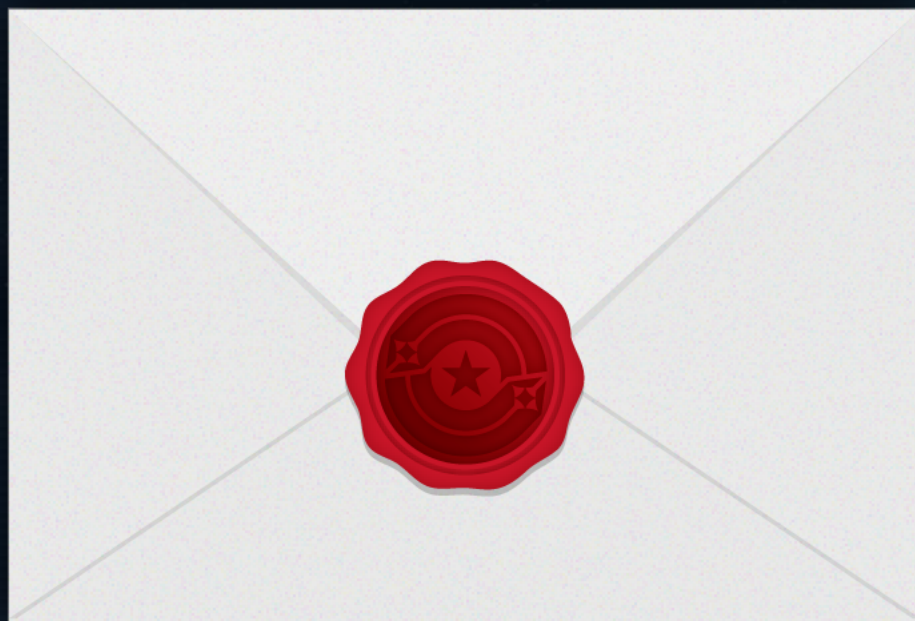
Exastro

 Search

Exastro

https://exastro-suite.github.io/docs/index_ja.html





contact@exastro.jp.nec.com

- クラウドネイティブのデモをお見せできます。
- ガイドブックを詳しく解説できます。

お問い合わせいただければと存じます



Exastro